# Practical No. 04

**Practical Name :** Install Kubectl and execute Kubectl commands to manage the Kubernetes cluster and deploy Your First Kubernetes Application.

**Aim :** To install Kubectl and execute Kubectl commands to manage the Kubernetes cluster and deploy Your First Kubernetes Application.

# Theory :

Kubectl controls the Kubernetes Cluster. It is one of the key components of Kubernetes which runs on the workstation on any machine when the setup is done. It has the capability to manage the nodes in the cluster.

Kubectl commands are used to interact and manage Kubernetes objects and the cluster. In this chapter, we will discuss a few commands used in Kubernetes via kubectl.

The Kubernetes command-line tool,kubectl, allows you to run commands against Kubernetes clusters. You can use kubectl to deploy applications, inspect and manage cluster resources, and view logs.

# Pods and Container Introspection Commands

|  |  |
| --- | --- |
| **Functions** | **Command** |
| Lists all current pods | Kubectl get pods |
| Describes pod names | Kubectl describe pod<name> |
| Lists all replication controllers | Kubectl get rc |
| Lists replication controllers in a namespace | Kubectl get rc – namespace=”namespace” |
| Shows the replication controller name | Kubectl describe rc <name> |
| Lists services | Kubectl get cvc |
| Shows a service name | Kubectl describe svc<name> |
| Deletes a pod | Kubectl delete pod<name> |
| Watches nodes continuously | Kubectl get nodes -w |

* **Debugging Commands**

|  |  |
| --- | --- |
| **Functions** | **Command** |
| Executes the command on service by choosing a container | Kubectl exec<service><commands>[-c<  $container>] |
| Gets logs from the service for a container | Kubectl logs -f<name>>[-c<  $container>] |
| Shows metrics for a node | Kubectl top node |
| Shows metrics for a pod | Kubectl top pod |

# Cluster Introspection Commands

|  |  |
| --- | --- |
| **Functions** | **Command** |
| To get version-related information | Kubectl version |
| To get cluster-related information | Kubectl cluster-info |
| To get configuration details | Kubectl config g view |
| To get information about a node | Kubectl describe node<node> |

* **Quick Commands**

|  |  |
| --- | --- |
| **Functions** | **Commands** |
| Launching a pod with a name and image. | Kubectl run<name> — image=<image- name> |
| To create a service detailed in  <manifest.yaml> | Kubectl create -f <manifest.yaml> |
| To scale the replication counter, counting the number of instances. | Kubectl scale – replicas=<count>rc<name> |
| Mapping the external port to the internal replication port. | Expose rc<name> –port=<external>– target-port=<internal> |
| Stopping all pods in <n> | Kubectl drain<n>– delete-local-data– force– ignore-daemonset |
| To create a namespace. | Kubectl create namespace  <namespace> |

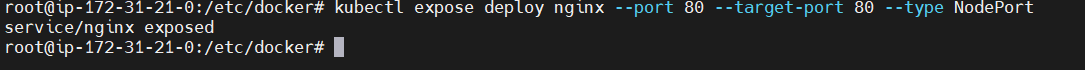
|  |  |
| --- | --- |
| To let the master node run pods. | Kubectltaintnodes –all-node- role.kuernetes.io/master- |

* **Steps to deploy application on Kubernetes cluster:** Execute the following command to create a deployment named nginx: kubernetes-master:~$kubectl create deployment nginx --image=nginx



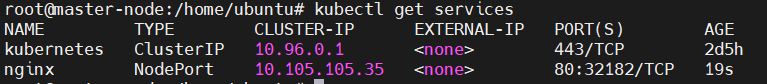
Next, run the following command to create a service named nginx that will expose the app publicly.

kubernetes-master:~$kubectl expose deploy nginx --port 80 --target-port 80 --type NodePort

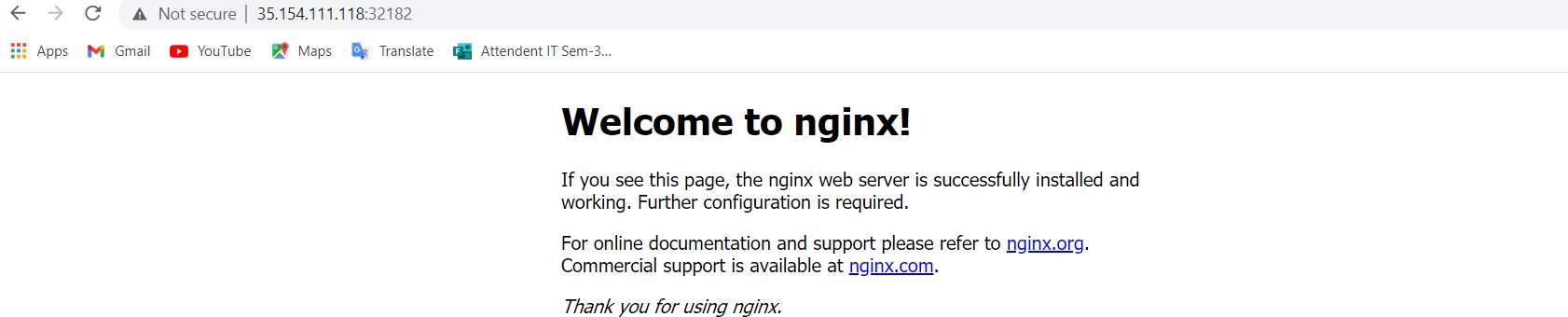


Run the following command:

kubernetes-master:~$kubectl get services



From the third line of the above output, you can retrieve the port that Nginx is running on. Kubernetes will assign a random port that is greater than 30000 automatically, while ensuring that the port is not already bound by another service.



To see the deployed container on worker node switch to worker01 on-slave#docker ps

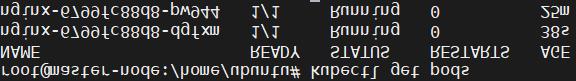


If you want to scale up the replicas for a deployment (nginx in our case) the use the following command:

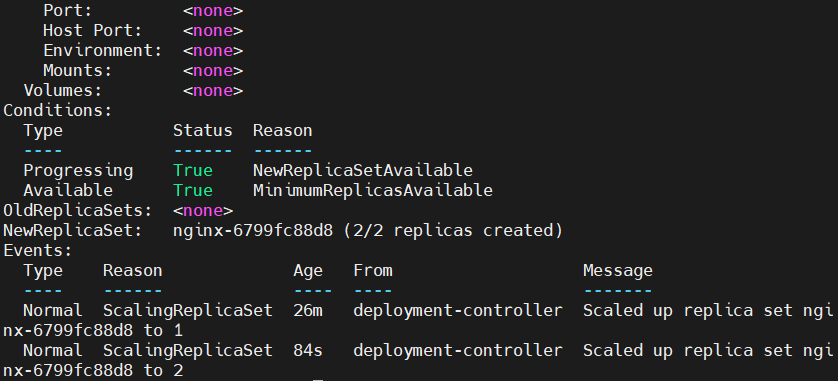
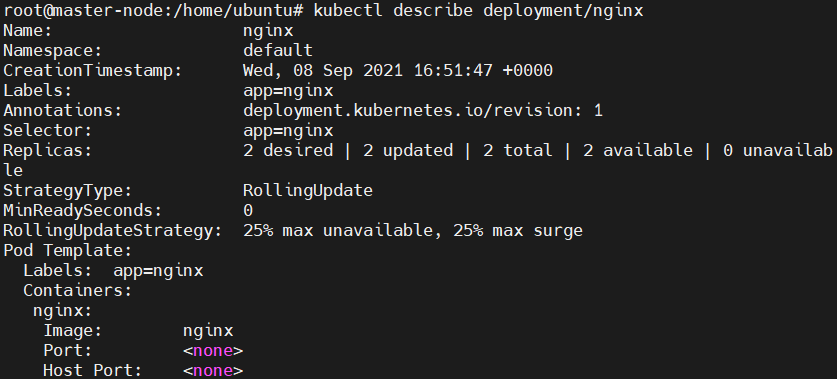
kubernetes-master:~$kubectl scale --current-replicas=1 --replicas=2 deployment/nginx



kubernetes-master:~$kubectl get pods



kubernetes-master:~$kubectl describe deployment/nginx



If you would like to remove the Nginx application, first delete the nginx service from the master node:

kubernetes-master:~$kubectl delete service nginx



Run the following to ensure that the service has been deleted: kubernetes-master:~$kubectl get services



Then delete the deployment:

kubernetes-master:~$kubectl delete deployment nginx



Run the following to confirm that this worked:

kubernetes-master:~$kubectl get deployments



# How to gracefully remove a node from Kubernetes?

On Master Node Find the node

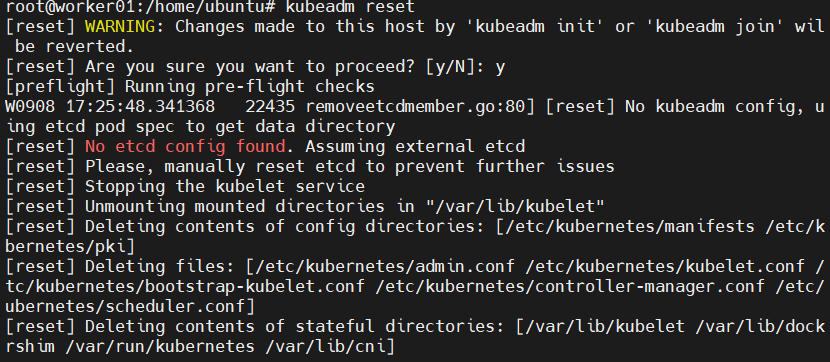
kubernetes-master:~$kubectl get nodes Drain it

kubernetes-master:~$kubectl drain nodetoberemoved Delete it kubernetes-master:~$kubectl delete node nodetoberemoved



On Worker Node (nodetoberemoved). Remove join/init setting from node kubernetes-slave:~$kubeadm reset

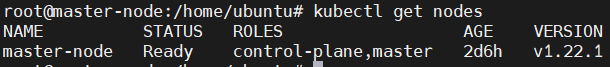
Press y to proceed



kubernetes-slave:~$docker ps



kubernetes-master:~$kubectl get nodes



# Conclusion :

Installed Kubectl and executed Kubectl commands to manage the Kubernetes cluster and deployed Kubernetes Application.